ElastiCluster

Automated provisioning of computational clusters in the cloud

Riccardo Murri <riccardo.murri@gmail.com>
(with contributions from Antonio Messina, Nicolas Bär, Sergio Maffioletti, and Sigve Haug)
What is ElastiCluster

ElastiCluster provides a **command line tool** to **create, set up and scale** computing clusters hosted on IaaS cloud infrastructures.

Main function is to get a compute cluster up and running with a single command.

Additional commands can scale the cluster up and down.
Example: SLURM cluster

Cluster definition is done in a INI-format text file.

```
[cluster/slurm]
cloud=openstack
login=ubuntu
setup=slurm
frontend_nodes=1
compute_nodes=4
ssh_to=frontend
security_group=default
image_id=...
flavor=4cpu-16ram-hpc

[setup/slurm]
frontend_groups=slurm_master
compute_groups=slurm_worker

[cloud/openstack]
provider=openstack
auth_url=http://...
username=***
password=***
project_name=***

[login/ubuntu]
image_user=ubuntu
image_user_sudo=root
image_sudo=yes
user_key_name=elasticluster
user_key_private=~/.ssh/id_rsa
user_key_public=~/.ssh/id_rsa.pub
```
ElastiCluster demo

1. Create 4 virtual machines on an OpenStack cloud
2. Install and configure a SLURM cluster on them
3. Connect to the cluster
4. Run an example
5. Add 1 more node to the cluster
6. Destroy the cluster when done

show time!
ElastiCluster features (1)

Computational clusters supported:
  ▶ Batch-queuing systems:
    ■ SLURM
    ■ GridEngine
    ■ Torque+MAUI
    ■ HTCondor
  ▶ Spark / Hadoop 2.x
  ▶ Mesos + Marathon

Distributed storage:
  ▶ GlusterFS
  ▶ HDFS
  ▶ OrangeFS/PVFS
  ▶ Ceph

Optional add-ons:
  ▶ Ganglia
  ▶ JupyterHub
  ▶ EasyBuild

(Grayed out items have not been tested in a while...)
Run on multiple clouds:

- Amazon EC2
- Google Compute Engine
- OpenStack

Supports several distros as base OS:

- Debian 7.x (*wheezy*), 8.x (*jessie*)
- Ubuntu 14.04 (*trusty*), 16.04 (*xenial*)
- CentOS 6.x, 7.x
- Scientific Linux 6.x
How does ElastiCluster work?

1. Create virtual machines in a cloud
   - done by Python code in ElastiCluster

2. Install and configure a pre-defined set of software
   - delegated to Ansible
ElastiCluster leverages Ansible to deploy and configure software:

- “playbooks” are list of idempotent tasks
  - playbooks can be re-run many times over
  - changes are *exactly reproducible*

- everything is on the client machine
  - no agent or bootstrap needed on cloud VMs
  - all configuration / playbooks hosted in a single place

- works on base OS images
  - independent from the cloud infrastructure
In a sense, ElastiCluster is just a large collection of Ansible playbooks.

But there is nothing special about these playbooks: any Ansible playbook can be applied by ElastiCluster

So, you can replace ElastiCluster’s playbooks, or add your own ones.
Issues

Setup time grows *linearly* with the number of cluster nodes.

Overcoming this seems to require a major change in how cluster setup is executed.

Ongoing discussion at:

https://github.com/gc3-uzh-ch/elasticluster/issues/365
Performance tip #1

To speed up setup, we need to reduce the amount of work that Ansible has to do:

1. create a prototype cluster;
2. make snapshots of each node type;
3. **create clusters using the snapshots** instead of the base OS image.
Performance tip #2

Ansible can run many playbooks in parallel.

But the default degree of parallelism is very conservative.

Increase the number of parallel connections!
A 1Gb/s network and a multicore CPU can easily accommodate a few 100’s parallel SSH connections.
Performance tip #3

Setup time grows with the number of nodes.

If you only care about CPU core count, use larger VM instance flavors.
Typical use cases

- **On demand provisioning** of computational cluster
- Clusters/servers for **Teaching**
- **Testing** new software or configurations
- **Scaling** a permanent computing infrastructure
On-demand provisioning of compute clusters

Google Genomics provides instructions to its users to spin up ephemeral GridEngine clusters.

“The instructions presented here are guidelines that have been used to create clusters up to 100 nodes. However when preemption rates are high, Elasticluster’s re-provisioning of clusters (via Ansible) often converges too slowly due to repeated failures.

For best success with the instructions here, it is recommended to keep cluster sizes to 20 compute nodes or fewer. For larger clusters, use regular (non-preemptible) virtual machines.”

Reference:

On-demand provisioning of compute clusters

Google Genomics provides instructions to its users to spin up ephemeral GridEngine clusters.

“The instructions presented here are guidelines that have been used to create clusters up to 100 nodes. However when preemption rates are high, Elasticluster’s re-provisioning of clusters (via Ansible) often converges too slowly due to repeated failures.

For best success with the instructions here, it is recommended to keep cluster sizes to 20 compute nodes or fewer. For larger clusters, use regular (non-preemptible) virtual machines.”

Reference:


Clusters for teaching

At UZH: JupyterHub+Spark clusters for teaching courses (e.g., data science) or for short-lived events (e.g., workshops).

At UniBas: make tiny “replica” clusters to teach new users.

Key ingredient is the ability to apply custom Ansible playbooks on top of the standard ones, to make per-event customizations.
Scaling permanent clusters

At UniBE: additional WLCG cluster for ATLAS analysis hosted on SWITCHengines

Scaling permanent clusters

At UniBE: additional WLCG cluster for ATLAS analysis hosted on SWITCHengines

“A 304 virtual CPU core Slurm cluster was then started with one command on the command line. This process took about one hour. A few post-launch steps were needed before the cluster was production ready. However, a skilled system administrator can setup a 1000 core elastic Slurm cluster on the SWITCHengines within half a day. As a result the cluster becomes a transient or non-critical component. In case of failure one can just start a new one, within the time it would take to get a hard disk exchanged.”

Any questions?

ElastiCluster source code:  
http://github.com/gc3-uzh-ch/elasticluster

ElastiCluster documentation:  
https://elasticluster.readthedocs.org

Mailing-list:  
elasticluster@googlegroups.com

Chat / IRC channel:  
http://gitter.im/elasticluster/chat
Credits

The initial ElastiCluster dev team:

- Antonio Messina
  <antonio.s.messina@gmail.com>
- Nicolas Bär <nicolas.baer@gmail.com>

...and the many users at UZH who, wittingly or not, have used ElastiCluster, reported bugs, and suggested improvements.